# TEACHING MOBILE COMPUTING USING CABANA

Paul E. Dickson
Hampshire College
School of Cognitive Science
893 West St.
Amherst, MA 01002, USA
(413) 559-5861
pdickson@hampshire.edu

**ABSTRACT**

Mobile computing is a growing market and an area of increasing interest to students. To answer our student demand we decided to develop a mobile computing course but one with a focus on interface and application design rather than implementation. The course is designed for a student audience ranging from novice programmers to experts. Using Cabana as our platform, we were able to achieve this end while also enabling our students to build functioning mobile applications. Cabana enables rapid interface design and application development without the learning curve of more traditional mobile application building. In this paper we describe our project-based curriculum and discuss its successes and failures.

## INTRODUCTION

The design of our mobile computing course at Hampshire College was largely influenced by the makeup of our student body and the backgrounds of the students likely to take such a course. We do not have predefined majors, and we have only two computer science professors. As a result our students who focus on computer science have a varied background in the field since they take many computer science courses at other campuses. Because of this we cannot offer a more traditional mobile computing course that expects students to be further along in their computer science education and able to rapidly pick up a new environment. In this paper we include no grading rubrics because Hampshire does not have grades, and notes on how we compiled our written evaluations would likely not prove useful to other educators.

We therefore decided to focus our mobile application design course on interface building and general application design, taking an application from the design board through usability testing. This skill set is useful for any level of computer scientist. This also enabled our more advanced students to do more ambitious projects without inhibiting our more general education and lower-level computer science students. We facilitated the development of the applications and their revision by running the course similarly to a studio arts course, with half of all class time (one day a week) spent having the class critique everyone's work.

In order to focus on design, we decided to use Cabana [1] to build our mobile applications. Cabana (then in beta testing) was developed by Department of Behavior and Logic (DOBL) and provides an environment that can create web-based mobile applications or device-based applications for iOS or Android [3].

## RELATED WORK

With the growing interest in mobile application development, the number of mobile computing courses offered has rapidly increased and papers have been written on the subject. The focus of many of these papers is quite reasonably on specific platforms and how to integrate them into a given campus environment. Rogers [10] and Ivanov [6] focused on developing for Apple's iOS and describe the rigors of getting courses that use iOS up and running on a college campus. They both discuss setting up the agreement with Apple to be able to develop on device for free and on the difficulties of developing courses for a moving target like the mercurial iPhone SDK. They also describe the necessity of learning Xcode, Objective-C, and Cocoa prior to iOS development, which was similar to our own experience [2].

Other papers focused on the environment used for mobile application courses [4, 5]. Fenwick et al. [4] argued that Android is the better platform because of Apple's making iOS a closed platform. Goadrich and Rogers [5] made more of an even argument about which platform is better. When they presented their paper, it became clear that the Mac user preferred iOS and the non-Mac user preferred Android, making their point that operating system preference had as much influence as anything else. Blackberry also has educational programs for schools and courses that use their hardware [9].

One theme common to these papers is that there is a learning curve for developing on whatever platform is chosen. The courses tend to be aimed at upper-level students who can more easily pick up a new environment and/or programming language. Kurkovsky [7] reinforced the point that this material is best suited for upper-level students when he described how developing mobile applications can help students gain experience working on computer networking, database management, etc., which are all higher-level concepts.

The entry point to mobile development does not have to be at a high level. Google's App Inventor [11] is designed to make mobile development accessible to a CS1-level audience. It uses a block programming interface and is easily learned. Unfortunately, this block language that makes it so accessible to new programmers can be frustrating to upper-level students who are comfortable writing code.

We chose to use Cabana [1, 3] because it provides an easy interface for newer programmers to learn while allowing advanced programmers to write more complex programs. It uses a wiring diagram approach for programming [3] but also allows developers to create their own modules that can be wired in. These custom modules can be written in JavaScript and can be as complex as the programmer desires. Cabana gave us the best combination of features to make our mobile application course accessible to multiple levels of students and that it would enable us to focus on design and testing rather than on learning SDKs for development. Cabana does not give the level of control of application development found with SDKs but enables the development of many complex applications.

One other option was to develop mobile applications that were web-based, and the Google Web Toolkit makes this quite feasible. Loveland [8] used this toolkit to enable students to build mobile applications in a human-computer interaction course in a manner similar to what we decided to do with our course. The advantage

we see in Cabana is that it can work for web development or for on-device development for iOS or Android.

## COURSE ORGANIZATION

The goal of this course is to teach mobile application development with an emphasis on design and usability. To this end the course was built around a series of application design projects. Each application was built over a series of homework assignments and design was facilitated through class discussions of student work. The course was divided into two major sections; the first was devoted to learning Cabana while the second related to building a single large application. An exact breakdown of material can be found at the course syllabus webpage http://helios.hampshire.edu/~pedcs/classes/cs227Spring11/syllabus.html. The course did not have a textbook.

### Learning Cabana

The first section of the course contained 2 of the 3 course projects. The first of these projects was related to implementing an old joke and was covered in 2 homework assignments while the second involved the building of a Tamagotchi (a virtual pet) and was covered in the next 3 homework assignments.

The old joke assignment relates to the joke that has floated around the Internet for years that involves a website that says that with the number of surveillance cameras in the world, all it needs is your name and date of birth to return a picture of you. After you put in your information and hit submit, it waits for a little while as if it is searching a database before returning a picture of a monkey, saying that it found your picture. This project taught students how to build a multi-screen application and import images into Cabana. It also gave them opportunities to start adding logic to their applications in order to make the joke work better. The second homework assignment had students adding to the complexity of the application based on newly learned Cabana programming and gave them an opportunity to make changes based on class comments about their application.

The Tamagotchi project functioned similarly. The 3 homework assignments covered 3 weeks and included 2 sessions in which students were able to give feedback on each other's designs. It also gave them an opportunity to implement more complex features in Cabana like flow components and persistent memory. This project was originally intended to run for only 2 assignments but students asked for it to be extended to 3 so that they could finish implementing changes based on feedback and have applications they could be proud of.

For both of these projects, students were given freedom regarding the exact details of the applications. As long as they followed the basic premise of the assignment and the work covered the features talked about in class, their work was accepted. This freedom seemed to inspire many of the students to push their boundaries and put in extra effort.

### Application Design

The meat of the course came in the second section and was covered by the third and largest project. This project was to build an application of the student's

choosing and to take it through all parts of the development process. They worked on this project for over half the semester and it covered 8 individual homework assignments.

The first assignment for students was to turn in 3 solid ideas for mobile applications that they would like to build. They were then required to discuss these ideas in class along with the pros and cons of each.

This was followed by a discussion of application context and application architecture design. For the assignment students had to pick which of their ideas they wanted to create and then describe the context for that application including usage location, intended audience, how it would be used, etc. They were also required to sketch the intended architecture of their application and include some sketches of what might appear on screen.

With context in mind, the course then moved on to paper prototyping and principles of interface design. The assignment for this was to build a physical paper prototype of their application for testing.

This led to discussions of what makes good applications. Students were also taught principles of simplifying interfaces for greater usability. The assignment here was to build a vertical slice of their application: a completely working section that showed all functionality.

With vertical slices in hand, students were then taught about developing strategies for their application marketing and how to promote their applications. The focus was on how they could make their applications more likely to be used. The assignment was to finish implementing their application.

The course then moved on to usability testing. For the assignment students were required to usability test their applications.

The usability testing results led to discussion of interface design in general. This discussion and the testing results were used in the next assignment when students where asked to revise their applications.

In the final week every application was presented and critiqued by the class. The final assignment was to implement any last suggestions and to pull together all of the information from the application creation process. It was a basic wrap-up of the course material.

**Lecture Overview**

The course was run in the mold of a studio art course, with one lecture a week devoted to critique of student work. Each of the 11 students presented material and had it critiqued for approximately 10 minutes of the 120-minute class. The length of presentation is not crucial and would have been shorter if there had been more students. This had a number of effects on the course and student production. To begin with it forced students to present orally every week and in doing so taught them how to present their ideas effectively. A couple of students mentioned that they had more public speaking experience from this course than from some that listed public speaking as a core part of the curriculum. This skill is generally overlooked in computer science courses.

The public speaking also led to a higher quality of work from many of the students. Fear of embarrassment during presentations convinced students to make a greater effort than they might have otherwise.

Improvement in application design was the greatest benefit of these presentations. Weekly feedback on their designs caused students to think more about the perceptions of people other than themselves when it came to functionality and layout. Everyone who writes software at times has focused solely on design from their own perspective and failed to consider how others would react to it. For many of the students this was their first exposure to this aspect of software development. The outside influences greatly improved student applications.

**Advanced work**

The design of this course made it possible for some students do more advanced work. Several students were frustrated that Cabana did not give the same level of device control they were used to from working with device SDKs. The course design enabled us to let these students develop applications using the SDKs while the rest of the course focused on Cabana. These students had to handle difficulties on their own and did so successfully.

The net effect was that these advanced students did more advanced work and there was no negative effect on the rest of the class. Both groups learned how to develop applications. Having students working on such divergent platforms was not difficult for us to grade because we give written evaluations. In a more standard grading system the same course could be listed with two separate course numbers, the higher one given to those students working with the device SDKs.

## USING CABANA

Cabana proved to be a good environment for the course but was not a perfect solution. Cabana was in beta testing for the entire duration of the course and this caused certain features to disappear during the semester or to start to work differently. Also, certain bugs appeared that students were unable to identify as bugs since they did not know whether they or the software had made a mistake since they were just learning it. On top of this the documentation for Cabana was very limited.

Despite these setbacks Cabana was a very good piece of software for developing mobile applications for the course. Students found it easy to learn and to use. Prototyping in Cabana proved extremely easy and student were able to rapidly develop and change applications. At the end of the semester only one student failed to have a working application and that was because some part of the back end of Cabana was changed in the last week of the class, causing code that had worked previously to stop working.

## CONCLUSIONS

We built a mobile application design course using Cabana that was successful for students with a variety of levels of programming experience. Cabana made it possible to focus on application design instead of just implementation, enabling our students to see development from a different angle. We were able to hold a lot of

class critiques that gave students feedback on their designs and presentations and greatly improved their skill set. These critiques would not have been possible if we had been required to teach an SDK.

On top of this, students were really pleased with the course and felt that they learned a lot. Many of them started looking at application design in new ways and it carried over to code they were writing for other classes. Their successful completion of mobile applications demonstrated their understanding of the course material. This Cabana-based mobile application curriculum is one that can work successfully at many institutions that also have students with varied programming backgrounds.

**ACKNOWLEDGMENTS**

We would like to thank DOBL for allowing us to use Cabana for this course and for providing timely feedback and development of requested features.

**REFERENCES**

[1] Department of Behavior and Logic Inc., Cabana - Create mobile apps with Cabana!, https://www.cabanaapp.com/landing/, retrieved November 15, 2011.
[2] Dickson, P. E., Experiences building a college video game design course, *J. Comput. Small Coll.* 25, (6), 104-110, 2010.
[3] Dickson, P. E., Cabana: A Cross-platform Mobile Development System, *SIGCSE '12: Proceedings of the 43nd Technical Symposium on Computer Science Education*, ?-?, 2012.
[4] Fenwick, Jr., J. B., Kurtz, B. L., Hollingsworth, J., Teaching mobile computing and developing software to support computer science education, *SIGCSE '11: Proceedings of the 42nd Technical Symposium on Computer Science Education*, 589-594, 2011.
[5] Goadrich, M. H., Rogers, M. P., Smart smartphone development: iOS versus android, *SIGCSE '11: Proceedings of the 42nd Technical Symposium on Computer Science Education*, 607-612, 2011.
[6] Ivanov, L., The I-phone/I-pad course: a small college perspective, *J. Comput. Sci. Coll.* 26 (6), 142-148, 2011.
[7] Kurkovsky, S., Engaging students through mobile game development, *SIGCSE Bull.* 41 (1), 44-48, 2009.
[8] Loveland, S., Human computer interaction that reaches beyond desktop applications, *SIGCSE '11: Proceedings of the 42nd Technical Symposium on Computer Science Education*, 595-600, 2011.
[9] Mahmoud, Q. H., Ngo, T., Niazi, R., Popowicz, P., Sydoryshyn, R., Wilks, M., Dietz, D., An academic kit for integrating mobile devices into the CS curriculum, *SIGCSE Bull.* 41 (3), 40-44, 2009.
[10] Rogers, M. P., Wrong number: avoiding the hidden perils in iPhone development, *J. Comput. Small Coll.* 25 (5), 300-305, 2010.
[11] Wolber, D., App inventor and real-world motivation, *SIGCSE '11: Proceedings of the 42nd Technical Symposium on Computer Science Education*, 601-606, 2011.